Applying Machine Learning Methods to Text Corpora and Case Bases (M.Tech Thesis)

APPLYING MACHINE LEARNING METHODS TO TEXT CORPORA AND CASE BASES

A Project Report
Submitted by

P. Deepak Subramoniam

In partial fulfillment for the award of degree of

MASTER OF TECHNOLOGY
in

COMPUTER SCIENCE AND ENGINEERING

Under the guidance of
Dr. DEEPAK KHEMANI

Department of Computer Science and Engineering
Indian Institute of Technology Madras
Chennai 600 036, India
May 2006

CERTIFICATE

This is to certify that the project work titled "**APPLYING MACHINE LEARNING TECHNIQUES TO TEXT CORPORA AND CASE BASES"** submitted by **P. DEEPAK** in partial fulfillment of the requirements for the award of the **Master of Technology in Computer Science and Engineering** is a record of bonafide work carried out by him in this Department. The context of this work, in full or in parts, has not been submitted to any other institute or University for the award of any degree or diploma.

Dr. DEEPAK KHEMANI

Associate Professor

Department of Computer Science and Engineering

Indian Institute of Technology, Madras

Chennai,

Date:

Acknowledgements

I take this opportunity to express my deep sense of gratitude to all those who helped me in completing this project.

The first and foremost factor responsible for the success of this project was the sincere direction and patience of **Dr. Deepak Khemani** under whose guidance this project was carried out. I express my gratitude and sincere thanks to him for giving me the opportunity to do this project under him.

I wish to thank **Prof. T.A Gonsalves**, Head Dept of Computer Science and Engineering and all the faculty members for their support and cooperation.

I wish to put on record, the guidance provided by **Dr. N. Narayanaswamy**, **Dr. B. Ravindran and Dr. Sankar Balachandran** for their valuable suggestions and critical inputs provided.

P. Deepak

Chapter One

Introduction

This project deals with the application of machine learning to text corpora to solve various problems of relevance to the text mining community. Additionally, we also attempt to improve case based classification systems. Clustering is among the most popular methods employed for unsupervised machine learning. Firstly, we attempt to cluster the vocabulary of a text corpus into clusters of words using partitional clustering techniques. Secondly, we use the word clusters thus formed along with statistical similarity metrics to label each document in the corpus by a small set of words. Thirdly, we move into an entirely different domain, namely URL corpora, and devise a similarity metric for URL pairs. Further, we go on to show that the similarity metric for URL pairs is representative of the similarities between the pages that the URLs represent by clustering standard corpora using the URL similarity metric. Fourthly, we utilize the URL similarity metric to find representative words for focused (topical) URL corpora[1]. We further show that the representative word-finder does give good performance even when in heterogeneous URL corpora. Fifthly, we investigate the potential of using data mining techniques to assign voting powers to cases in a case based classification task, and investigate the utility of such techniques in the specific task of spam filtering.

As most of the tasks that we have attempted are largely independent of each other, we devote separate chapters to each of the tasks. Chapter 2 deals with the task of word clustering. Chapter 3 deals with the application of word clustering for automatic labeling (or indexing) of documents. Chapter 4 deals with the similarity metric between URL pairs, clustering of URL corpora and the task of finding representative words for URL corpora. Chapter 5 deals with the application of differential voting on a case-based classification system. Chapter 6 lists the outputs of the project so far.

[1] Topical or Focused URL Corpora are those corpora of URLs of web pages, most of which relate to a particular topic

Chapter 2

Building Clusters of Related Words from a Corpus: An Unsupervised Approach

1. Introduction

This study focuses on building sets of semantically related words from a corpus of documents using traditional data clustering techniques. The task of building semantically related sets of words from a corpus of documents and allied problems have been studied extensively in the literature. Most of these techniques stem from the Computational Linguistics community and many involve parsing of the documents. We represent each word as a vector which reflects the distribution of occurrences of the same word in the different documents. Semantically related sets are derived by means of clustering of these word vectors. Our work presents a significant departure from the earlier literature in dealing with the problem. Firstly, we do not make use of any parsing or part of speech tagging techniques and represent each document as just a bag of words, a common representation in the information retrieval and data mining community. Secondly, we attempt to use the information based on the document frequencies of the words whereas the traditional approaches have treated the entire corpus of documents as a collection of sentences. Once again, document frequency has been put to good use and comes from the data mining community [1]. Thirdly, given that our aim is to build collections of words, we use the k-means clustering algorithm instead of hierarchical agglomerative clustering which has been the popular choice in literature.

2. Related Work

One among the earliest works which focuses on a related problem [3] talks about identifying the ranked list of similar nouns, given a noun in the corpus. It introduces the concept of mutual information for a noun-verb pair and extends it to define a similarity

measure for every noun-noun pairs. The more similar they are according to the similarity measure, the more semantically related, they are expected to be. [6] addresses the problem of clustering words to find sets of similar words. Words are represented by the relative frequency distributions of contexts in which they appear, and relative entropy is used to measure the dissimilarity of those distributions. A soft hierarchical clustering of data is done to get the relationships between the words. [2] describes a methodology to create a thesaurus from a given corpus. From each sentence, they derive triples of the type <w1,r,w2> which indicates that word w1 is related to word w2 by the relationship r. Such triples are used to derive a similarity measure for word-pairs which quantifies the confidence that a word describes another. Hierarchical Agglomerative Clustering is done to get a tree structure to represent the entire thesaurus. [4] addresses the problem of finding the hypernyms of a particular noun (similar to [2]). The similarity measure of a noun pair is parameterized only by the number of times the nouns co-occur in a conjunction or appositive with the other in contrast to [2]. Hierarchical agglomerative clustering is done to obtain a tree which is further used to find hypernyms. As can be seen, most of the work in this regard comes from the computational linguistics community. Finding 'semantic relationship' is almost always considered as a problem which involves parsing and exploitation of co-occurrence information. Further, all the methods referred to above, use hierarchical agglomerative clustering to build a measure of semantic relationship between words. [21] uses LSA (Latent Semantic Analysis) and PoS tag information for finding related words. LSA uses Singular Value Decomposition (SVD), a dimension reduction technique, which brings related words closer in the reduced dimension. While it is possible to use LSA for clustering related words, it has been used in conjunction with PoS tag information.

3. Motivation

Clustering is a very popular technique in the data mining community and has been applied to document collections to find clusters of similar documents. It has been shown [8] that standard k-means clustering works better than hierarchical techniques for document clustering. k-means has been very popular in the text mining community and

many variants have evolved over time (of which most of them try to incorporate semi-supervision) such as COP-k-means [9], Seeded k-means and Constrained k-means [10]. Given that text clustering using k-means has worked well, it is obvious that it has been able to infer the semantic relationship between the different documents. We outline a very standard methodology for extracting the vectors from the text corpus and try to put forward our motivation in a simple and intuitive manner.

Given a corpus, the text clustering task usually starts off with building the term document matrix which has as many rows as the number of documents and as much columns as the number of words. Each entry in the matrix indicates the number of times the corresponding word has occurred in the corresponding document. Each row corresponds to the TF (term frequency) vector of the particular document. Further techniques to process the matrix involve normalization of each document vector to add up to a constant, whereby we get the normalized TF (nTF) vector. An additional step of Inverse Document Frequency Weighting may be incorporated before normalization, whereby we get the normalized TF-IDF vector. Given the TF, nTF or nTFIDF vectors of the documents, clustering is a straightforward task. Having outlined the document clustering task, an analogous method of term clustering is not very difficult to perceive. In the term document matrix, each column corresponds to a term and the transpose can be used as a Document Frequency (DF) vector, whereas a normalized version of the DF vector could analogously be termed the nDF vector.

Given that the clustering of TF, nTF and nTFIDF vectors do aid discovering semantic relationships among documents, we argue that the clustering of the DF, nDF and nDFITF vectors would aid discovering semantic relationships among the words. More abstractly, we argue that if the clustering of the rows of the term document matrix is useful, clustering of the columns of the term-document matrix can't be useless. Having put forward our motivation, we go forward to verify and quantify the utility of such an approach.

4. Experiment Methodology

We used the Time corpus [11], a popular dataset in the Information Retrieval and Data Mining communities which consists of 423 articles published by the Time magazine during the cold-war period (1960s). The corpus consists of documents which have comparable lengths which is a desirable property for our experiments. The entire dictionary of words in the corpus, after stop-word removal and stemming, is of size 20000. As already mentioned in section 3, we represent each term by the corresponding column vector from the term document matrix (whose elements are term frequencies in a document). The raw column vectors normalized so that each vector has elements summing up to unity from the term document matrix are hereafter referred to as the normalized Document Frequency (nDF) vectors. We use the set of nDF vectors in our experiments. The work makes no assumption on the type of clustering methodology to use. However, for the purpose of this experiment we demonstrate results using k-means clustering. The sets of nDF vectors were clustered using the k-means algorithm. k-means takes the number of clusters (to be generated in the output) as a parameter. Given that we do not have any knowledge of the number of clusters that exist, we tried out different values of k. The values chosen were 10, 20, 50 and 100. The actual k-means clustering was done using the WEKA Toolkit for Data Mining [12] developed by the University of Waikato, New Zealand. With a dictionary size of 20000, the average cluster size is of the order of hundreds of words. Evidently, it is difficult to manually verify the goodness of the clusters. One obvious solution would be to compare the clusters with a well-defined ontology such as WordNet [13]. But the Time corpus had a lot of proper nouns such as names of countries and people who were in the news during the cold-war period, thus rendering the comparison with WordNet inappropriate. We introduce a hypothesis which aids us in evaluating the clusters.

Hypothesis: The points (words) closest to the cluster center are representative of the cluster.

As this hypothesis is intuitively justifiable to an extent, we choose not to further explain it here. We take the m closest points to the cluster center (for each of the clusters

generated) and use them to represent the cluster. We call these words as "Representative Words." If the words thus selected are semantically related to each other then the representative words can be said to be semantically coherent. As we move away from the cluster center, the semantic coherence of the words is expected to decrease. For the purpose of our experiments, we take five words (m=5) closest to the cluster center. To evaluate the semantic coherence of the five words thus selected, we use independent knowledge sources such as Google [14] and Wikipedia [15]. Past work, including, [18], [19] and [20], all compare their results against a lexical resource like WordNet. Instead, we appeal to knowledge sources like Google and Wikipedia to evaluate the effectiveness of our clusters. We queried the knowledge sources manually using various combinations of the five words of each cluster and tried to understand the semantic similarity among them in the context of the cold-war (the timeframe during which the articles in the time corpus were published).

5. Results

We present herewith the results of the experiments. We choose to present the results of the experiments with k=10 fully and a sample of semantically coherent clusters from the results for k=50 (due to space limitations). Some clusters have less than 5 words in them. Descriptions are not provided for clusters whose semantic coherence we were unable to mine manually and those for which the semantic coherence is evident. The representative words are ordered in ascending order of distances to the center of their clusters. All descriptions were derived by summarizing information found using Google and Wikipedia and do not represent the authors' opinions about the word cluster. All results in the tables that follow have been gathered with m=5. To get a feel of the decrease in semantic coherence with increasing m, we present some representative results for m>5 herewith.

Table 1. Representative Results with m > 5 for clusters gathered with k = 50, words ordered in the order of increasing distance from the centroid of the cluster

<Brunei, borneo, Malayan, malay, Singapore, Malaysia, Indonesia, sukarno, malaya, federation,

rahman, abdul,…>
<Syria, middle, Arabs, Syrian, Unity, Jordan, Saudi, Union, Iraq, Aflak, Egypt, Yemen, Baath, Arab, Nasser…>
<elisabethville, leopodville, united, central, congo, Katanga, tshombe, troops, president, police,…>

5.1 k = 10

Table 2. Results for the application of k-means with k = 10

Cluster #	Representative Words (m=5)	Descriptions from Independent Knowledge Sources
0	Damascus, Arabs, Syrian, Egyptian, Jordan	**Syria**, **Egypt** and **Jordan** are **Arab** nations. **Damascus** is the capital of **Syria**
1	time, minister, years, labor, week	**Labor** is a political party which had **ministers** in power during the cold war **years**
2	European, charles, nuclear, market, french	**French** are the peoples of the **European** nation of France
3	lemass, Ireland, irish, Dublin	**Ireland**, whose peoples are called **Irish** has its capital at **Dublin**. Sean **Lemass** was an Irish political leader
4	Saigon, Vietnam, cong, Buddhist, nhu	**Saigon** is district one of ho-chi-min city, the capital of **Vietnam**. Madame **Nhu**, the first lady, was a member of the Viet **Cong**, which had anti-**Buddhist** policies
5	small, including, finally, high, united	
6	Brunei, malay, Malayan, borneo, Singapore	**Malay** is the language spoken by the **Malayan** people and is the official language of Malaysia, **Brunei** and **Singapore**. The Malaysian city of Sabah was called British **Borneo** when it was a British colony
7	constantly, ability, mistakes, endless, aide	
8	peking, red, mao, soviet, communist	**Peking** was the former name of the Beijing, the capital of china where the book called the little **red** book of quotations by **Mao** Zedong was published in 1962. he was trying to drive a wedge between Moscow of **soviet** Russia and Peking of China. Both

China and Soviet Russia were **communist** nations.

9	famine, densely, Malthusian, ecological, bachelors	

5.3k = 50

Table 4. Representative Results for the application of k-means with k = 50

Cluster #	Representative Words (m=5)	Descriptions from Independent Knowledge Sources
0	white, African, Africa, black, god	**Africa** is known for the racist turmoil between the **white**s and the **black**s.
30	tents, Libyan, fires, mosques, bricks	**Libya** is known for its **mosques**.
35	Pakistan, India, Kashmir, Nehru	**Nehru** was the prime minister of **India**, which has a dispute with neighboring **Pakistan** over the occupancy of **Kashmir**
44	sinistra, palmiro, Giovanni, toligatti, leone	Partito Comunista Italiano, the Italian Communist Party was headed by **Palmiro Toligatti**. **Giovanni** was an Italian astronomer. **Leone** Battista was an Italian painter.

6. Conclusions

Firstly, as the results show, the clustering of words does indeed reveal the semantic relationship among the words in the corpus. There is a definite bias due to the corpus used, which in this case is the bias of the cold war period i.e., the semantic relationship between the words in the cold war period is being revealed through our experiments. The results confirm our hypothesis that clustering of nDF vectors would reveal the semantic relationships. Secondly, as can be seen from the results, the proper nouns (such as India, Singapore, Malay) and their variations (such as Indian, Malayan etc.) get separated from the common nouns i.e., the set of representative words for a cluster is seldom a mixture of proper nouns (and their variations) and proper English words. There are exceptions such as the cluster #9 for k=10. In general, unless there is a strong semantic relationship, the proper nouns get reasonably well separated from the other English words. Thirdly, the semantic relationship between proper nouns (and

variants) is made explicit by clustering. The problem of identifying the semantic relationships between proper nouns has been well studied in the computational linguistics literature. Here, we have achieved a reasonable accuracy of identifying the semantic relationship between nouns without using either parsing or part-of-speech tagging, which are considerably expensive and often used in the literature. The clusters which have only proper nouns in them, such as <Pakistan, India, Kashmir, Nehru> testify our claim. Fourthly, the increase in distances of the representative words from the center of the cluster does present a decreasing amount of semantic coherence with the words closer to the center. For example, the inclusion of Singapore (word #5) into the top 4 words in cluster #6 with k = 10, does decrease the coherence of the set. Another example would be inclusion of Nehru (word #4), the name of a person into the top 3 words (Pakistan, India and Kashmir) which are names of places and nations. It is interesting to observe that the ordering of words does convey some clues to the semantic relationships.

6. Using Clusters of Semantically Related Words

Clusters of semantically related words could be used for query expansion and query relevance measurement in an information retrieval (IR) system. Words in the same cluster as the query words for a query posed by a user in an IR system can be used as suggestions to expand the query for better retrieval. In a multi-word query, whether or not the query terms appear in the same cluster could be used to measure the relevance of the query to the corpus in an IR system. For example, “Moscow+nuclear” would be a less relevant query (to the Time corpus that we have used in our experiments) compared to “France+nuclear” as the query terms in the former appear in different clusters and the query terms in the latter appear in the same cluster.

The technique could be used in a variety of corpus-based unsupervised learning tasks. The hypothesis that words close to the center are representative of the cluster can be used to identify the topic that a topical corpus (a corpus that deals with a specific topic) deals with, by considering the entire collection of words in the corpus as a single cluster. Topical corpora include the collection of postings in a forum, of chat sessions in a

focused chat room, and that of entries in a topical weblog. Identification of the context (or sense) of a term's usage in the corpus can be done by means of the semantically related word clusters. For example, "red" is used in the context of the Moscow and Soviet Union rather than in the context of colors as "Moscow" and "Soviet" are in the same cluster as "red" in our experiments. Similarly, "cong" is used in the context of the Vietnam Congress and not the US congress.

Identification of semantically related word clusters would aid in automatic annotation of documents from the space of the entire vocabulary (as opposed to classification tasks which have a fixed small set of labels). Each document could be assigned to one or more clusters (based on statistical similarity measures) and the most representative words from those clusters could be used to label the document. Note that in such an approach, the label of a document need not necessarily come from the set of words present in the document. Such sets of words may well be used as compact representations of documents for data mining tasks. This is further detailed in the next section, which deals with this precise problem.

The technique presented is general enough that we could replace the set of words by a set of features and identify the semantic relationships between features. For instance, a spam filter may use features such as occurrence of phrases (e.g., "over 21", "mortgage rates") and other non-trivial features such as background color (a red background color is indicative of a porn mail). We would expect our technique to cluster features specific to a category of mail together, e.g., for instance features specific to porn mail might just fall into the same cluster.

As our results show, most of the representative words for a cluster are determiners for particular classes. For instance, the most representative words of cluster 0 in table 2 would intuitively be good determiners for documents relating to the Arab world i.e., documents relating to the Arab world would cluster well together if we project them on the space of the 5 most representative words of cluster 0. Given that each set of most representative words would most likely be good determiners for one category or the

other, the projection of the documents on the union of all such sets would separate out the documents based on the class they belong to. Thus, taking the union of all most-representative-word sets could be used as an unsupervised feature selection technique for document clustering.

8. Contributions and Future Work

Firstly, we have demonstrated nDF vector clustering as a feasible tool for the extraction of semantically related sets. Secondly, by means of our hypothesis that words closest to the cluster center are representative of the cluster, we have proposed a means of evaluating cluster quality even for a large number of clusters. Thirdly, this is the first study which tries to extract semantic information using the bag-of-words model for documents without using any linguistic techniques. Fourthly, to the best of our knowledge, this is the first study which verifies the applicability of the low-cost k-means algorithm for term clustering. All earlier studies have used the more expensive hierarchical agglomerative clustering algorithm. It is intuitive that the value of k is hardly predictable in cases such as clustering on text data. We experimented with varying values of k in our experiments. Future work could use techniques such as Bayesian Information Criterion [16] to estimate the number of clusters or use algorithms such as bisecting k-means [17] which don't require k as an input parameter. Further, as mining semantic relationships between nouns is the most interesting component of extracting semantic information, computational linguistics techniques could be used to find nouns in the dictionary and cluster them alone. This would render the technique comparable to the techniques which aim at finding the semantic relationship between nouns (from the computational linguistics community). Further, nDFITF clustering could also be tried out to get further insights about the distribution of words in text corpuses.

References

1. Harry Wu, Gerard Salton, “A comparison of search term weighting: term relevance vs. inverse document frequency”, Proceedings of the 4th Annual ACM SIGIR, Californian, 1981, pp.30-39
2. Dekang Lin, “Automatic retrieval and clustering of similar words”, Proc of COLING-ACL, 1998, pp.768-774
3. Hindle, “Noun Classification from predicate argument structures”, Proc of the 28th Annual Meeting of the ACL, 1990, pp. 268-275
4. Caraballo, Charniak, “Automatic construction of a hypernym-labeled noun hierarchy from text”, Proc of the 37th Annual Meeting of the ACL, 1999, pp.120-126
5. A. Maedche, V. Pekar, S. Staab, “Ontology learning part one - On discovering taxonomic relations from the web.” In Web Intelligence, pages 301-322. Springer Verlag, 2002.
6. F. C. Pereira, N. Thishby, L. Lee, “Distributional Clustering of English Words”, In Proc of the 30th Annual Meeting of the ACL, 1993, pp.183-190
7. M Sanderson, W V Croft, “Deriving Concept hierarchies from text”, Proceedings of the 22nd SIGIR Conference, 1999, pp.206-213
8. Steinbach. M, Karypis. G, Kumar. V, “A comparison of document clustering techniques”, In Proceedings of the KDD Workshop on Text Mining, Boston, 2000
9. K. Wagstaff, C. Cardie, S. Rogers, S Schroedl, “Constrained K-Means Clustering with background knowledge”, Proceedings of the 18th International Conference on Machine Learning (ICML-2001), pp. 577-584
10. Sugato Basu, Arindham Banerjee, Raymond Mooney, “Semi-supervised Clustering by seeding”, Proc of the 19th Intl Conference on Machine Learning (ICML), 2002, pp. 19-26
11. Sabine Bergler. “Collocation patterns for verbs of reported speech--a corpus analysis oil tile time Magazine corpus”. Technical: report, Brandeis University Computer Science,. 1990.
12. “Weka: Open Source Software for Data Mining”, www.cs.waikato.ac.nz/ml/weka/
13. “WordNet: Online Lexical Reference System”, http://wordnet.princeton.edu
14. Google, http://www.google.com
15. Wikipedia, The Free Encyclopedia, http://en.wikipedia.org

16. Dan Pelleg, Andrew Moore, “X-means: Extending K-means with Efficient Estimation of the Number of Clusters”, Proceedings of the 7th International Conference on Machine Learning (ICML), 2000, pp. 727-734
17.Sergio N Saveresi, D L Boley, “A comparative analysis on the bisecting K-means and the PDDP clustering algorithms”, Intelligent Data Analysis Journal, 2004
18. E. Riloff, J. Shepherd, “A corpus based approach for building semantic lexicons”, In Proceedings of the Second Conference on Empirical Methods in Natural Language Processing (EMNLP), 1997, pp.117-124
19. Brian Roark, Eugene Charniak: “Noun-Phrase Co-Occurence Statistics for Semi-Automatic Semantic Lexicon Construction.” In Proc. Of the 36th Annual Meeting of the ACL, 1998, pp.1110-1116
20. Dorow. B, Widows. D, “Discovering Corpus-Specific Word Senses”, Proc. of the 10th Conference of the European Chapter of the ACL, 2003, pp.79-82
21. Dominic Widdows, Unsuperivsed methods for developing taxonomies by combining syntactic and statistical information, HLT-NAACL 2003

Chapter 3

Corpus Based Unsupervised Labeling of Documents

1. Introduction

The World Wide Web and the Internet has resulted in an explosion of data sources like web pages, weblogs, newsfeeds and email to name a few. These documents span over a wide range of topics and are very dynamic in nature. Text categorization can be used as a tool to organize and manage this data. The dynamic nature of these data sources make it difficult to define a closed set of labels that could be assigned to documents. One approach to tackle this would be to assign labels using a few important keywords from the document. For example, the articles on weblogs could be labeled so that they are served to a wider audience. The current method optionally makes use of the "tags" feature where labels are manually assigned by the blog writer. Similarly web search results can be categorized for efficient browsing. Traditional, hand labeled techniques for text categorization makes it impossible to handle such copious data. Besides most manual techniques are laborious and error prone. Several methods have been suggested in the past to alleviate the labeling problem. Many of these methods rely on the availability of some kind of training data, building a classifier and using the classifier to further label the unseen data. Training data might be sparse or difficult and expensive to obtain. Given the wide scope of the documents on the web and their dynamic nature it is not possible to rely on a model that has been trained on a single corpus. As [8] have noted, the diversification of applications of automatic text categorization makes it difficult create training data for each application area. Attempts have been made to reduce the amount of training data like using a combination of labeled and unlabeled data. We review some of these methods in the following section.

In this work we propose an unsupervised attempt to labeling documents. We use the traditional bag-of-words representation of text and represent each word as a vector which reflects the distribution of words in the different documents. So this method can be readily incorporated in any of the existing IR systems that use the same representation.

These word vectors are then clustered using the k-means [2] clustering algorithm. We draw a set of representative words from each cluster as a label and derive a set of candidate labels. A label from the set of candidate labels is assigned to each document that maximizes the norm of the dot-product of the document vector and the label vector. The method presented here is significantly different from the previous works as it does not require any manual intervention or labels.

2. Related Work

An entire gamut of machine learning techniques like supervised, semi-supervised and unsupervised learning has been applied at various levels to the task of text categorization. We review some of these techniques and contrast how our work differs from them.

The supervised selection techniques rely on the presence of training data. The training data is usually in the form of a few labeled documents. A classifier is trained from these labeled documents is used for further classifying of unseen documents. Work done by ([3] & [4]) using Naïve Bayes classifiers, [5] using support vector machines, [6] using classifiers based on the MDL principle, [7] using probabilistic models and by InfoSeek using neural networks. Although these methods perform well they require training data which might be difficult to obtain. The problems with manual labeling resulted in development of semi-supervised techniques by ([8], [9], [10], [11], [12], [13], [14], and [15]). These methods are characterized by the use of both labeled and unlabeled data. The above methods, viz, supervised and semi-supervised learning make use of labeled training documents, although in differing quantities. Our approach differs from these as we do not make use of labeled training data. Two earlier works ([16] & [17]) use unsupervised methods to text categorization. The former [16] makes use of labeled words instead of labeled documents. They expect the user to provide a few "representative words" for each class and use this information along with the clustered results to build a document classifier. Our method differs from this as we do not take any additional input from the users apart from the unlabeled corpus. Another study [17], on the other hand

create “training sentence sets” using keyword lists of each category and use them for training and classifying text documents. This scheme, as a part of its preprocessing step, derives features by part-of-speech tagging of the text. We do not make use of such features.

3. Proposed Approach

Broadly, our approach can be divided into four sequential phases, as below. In the following subsections, we go on to describe each of the different phases in greater detail.

• Clustering of Words to Arrive at Semantically related Word Clusters
• Generating lists of representative words for each Semantically Related Word Cluster
• Tagging documents with clusters
• Building labels for each document from the clusters it is tagged with

We reuse the methodologies devised in Chapter 2 to do the initial two subtasks. The methodology used is exactly the same as in Chapter 2 for those subtasks. We go on to describe the other subtasks in greater detail by means of the subsections below.

3.1 Tagging Documents With Clusters

Having arrived at a concise representation for each cluster of related words, this phase starts off with assigning a score for each document-cluster pair. The following formula computes the score for the <d,C> pair, where d is a specific document and C is the set of representative words for a specific cluster.

$$\text{Score}(d,C) = \sum_{c \in C} \text{(frequency of c in d)}$$

As can be seen, it just computes the sum of frequencies of each word in C, in the document d. This is done for every document in the corpus (used in the first phase) and every word cluster (generated in the first phase). Thus, for each document, we have an array of scores, with one entry per cluster. We choose to map a document to the cluster(s), with whom, it has the highest score, provided the highest score is greater than zero. Thus, if there is a tie and a document has multiple highest scores, all the clusters with the highest scores are taken to tag the document. Further, it may be noted that a

document may not be tagged with any clusters, if it has no occurrences of any of the representative words in any cluster. We expect that such a case would be very rare. At the end of this phase, we have each document tagged with clusters. This phase is represented in pseudo-code as shown in Table 1.

Table 1. Algorithm for tagging documents with clusters

```
Tag_Doc_With_Clusters(Corpus COR, Clusters CL)
{
    for each document d in Corpus COR,
    {
        for each Cluster C in CL, Score(d,C) = Σ_{c∈C} (frequency of c in d)

        Clusters to tag d with is defined as
          { C ∈ CL | (Score(d,C) > 0 && Score(d,C) >= Score(d,C1)) ∀ C1 ∈ CL }
    }
}
```

3.2 Building Labels for documents

This phase assigns labels (a word or multiple words) to each document in the corpus. The label would always be a subset of the union of the representative words of the clusters with which the document is tagged. A cluster may get a high score with a document if one of its representative words occurs very frequently in the document, even if none of the other representative words occur in the document at all. This phase smartly shields against such hostile cases. For each cluster that a document is tagged with, the average number of occurrences (in the document) of words in the representative word-set is computed and all words (among the representative words) which have at least as many occurrences as half of the average so obtained, are added to the label of the document. Consider a hostile case where, among the set of representative words {p, q, r, s, t} for a cluster which occurs among the tags of a document d, word ‘p’ occurs 100 times in d (inducing a score of 100) and all others do not have any occurrences. Only ‘p’ would be added to the label as none of the other words have more than 10 occurrences, 20 being the average number of occurrences for that cluster-document pair. Although the

algorithm is prone to less hostile cases, our results reaffirm that half the average number of occurrences is a good enough threshold. The algorithm is represented in pseudo-code as shown in Table 2.

Table 2. Algorithm to Build Labels for Documents

```
Build_Label_for_Documents(Corpus COR, Clusters CL)
{
    for each document d in Corpus COR,
    {
        Label(d) = Ø
        for each Cluster C among the tags of d,
        {
            Average_Score(d,C) = Score(d,C)/|C|
            Label(d) = Label(d) ∪ {c ∈ C | (frequency of c in d) > (0.5*Average_Score(d,C))}
        }
        Output <d,Label(d)>
    }
}
```

4. Experiments and Results

We chose to test our approach on the Time corpus [18], a popular dataset in the Information Retrieval and Data Mining communities which consists of 423 articles published by the Time magazine during the cold-war period (1960s). Recall that this is the same dataset used in chapter 2 to illustrate the feasibility of word clustering. The entire dictionary of words in the corpus, after stop-word removal, is of size 20000. We chose not to use the labeled corpuses popular in literature as those corpuses mostly had very abstract labels whereas out approach generated very specific labels. For instance, the article which talks about Indian Prime Minister Jawaharlal Nehru's talks with Pakistan counterparts on the Kashmir issue would most probably, be labeled just "Kashmir" in a labeled corpus, whereas our approach generates "India" , "Pakistan" , "Kashmir" , "Indian" and "Nehru" as labels. Further, manually assigned labels tend to have words not in the document. Just to cite an example, an article on a company buying stakes in

another company would most probably be labeled “acquisition” whereas our approach can, at best, come up with “buy”, “stakes” among the labels.

We proceed to illustrate the labeling of documents that we arrived at using our approach. We present the <document name, extract from document contents, labels, score> triplets for a random sample of the results from our experiments. Due to space constraints, we present the results of our experiments in Table 4. where k was set to 100 in k-means.

Table 3. Results of Labeling Documents

Document Name: TIME071 ***Extract from the document:*** … EUROPE A NEW & OBSCURE DESTINATION IN AN ALLIANCE IN WHICH PARTNERS HAD BECOME INCREASINGLY MINDFUL OF ONE ANOTHER'S SENSITIVITIES, IN WHICH VICTORIES WERE TACTFULLY NOT CROWED OVER, AND TOGETHERNESS IN ITSELF WAS REGARDED AS A GOOD THING, CHARLES DE GAULLE LAST WEEK REMINDED THE WORLD OF WHAT ONE … ***Labels:*** **gaulle,france,europe,de** ***Score:*** 168
Document Name: TIME370 ***Extract from the document:*** … IN 1845, BEFORE THE POTATO FAMINE DECIMATED ITS POPULATION, IRELAND WAS WESTERN EUROPE'S MOST DENSELY SETTLED COUNTRY; SINCE THEN, ITS 9,000,000 INHABITANTS HAVE DWINDLED TO 2,824,000 . IRELAND IS THE ONLY NATION IN EUROPE WHOSE POPULATION HAS SHRUNK IN THAT TIME . WHILE IRISHMEN LEFT THE COUNTRY IN WAVES, THEY ENTERED IT … ***Labels:*** **ireland,irish,lemass** ***Score:***135
Document Name: TIME024 ***Extract from the document:*** … KASHMIR TALKING AT LAST THE BRITISH RAJ, WHICH ONCE CONTROLLED INDIA'S NORTHWEST FRONTIER PROVINCE OF KASHMIR, EXACTED A TOKEN ANNUAL TRIBUTE OF TWO KASHMIRI SHAWLS AND THREE HANDKERCHIEFS FROM THE MAHARAJAH . NEVER SINCE HAS THE PRICE OF PEACE BEEN AS SMALL . IN THE YEARS AFTER INDEPENDENCE IN 1947 SPLIT THE INDIAN SUBCONTINENT … ***Labels:*** **indian,Pakistan,india,kashmir,nehru** ***Score:***64

Document Name: TIME464

Extract from the document: … SOUTH VIET NAM REPORT ON THE WAR OVERSHADOWED BY THE POLITICAL AND DIPLOMATIC TURMOIL IN SAIGON, THE ALL BUT FORGOTTEN WAR AGAINST THE VIET CONG CONTINUES ON ITS UGLY, BLOODY AND WEARISOME COURSE . THE DRIVE AGAINST THE COMMUNISTS HAS NOT DIMINISHED IN RECENT WEEKS ; IN FACT, IT HAS INTENSIFIED . FEARS THAT THE …

Labels: **Vietnam,south**

Score: 50

Document Name: TIME381

Extract from the document: …COMMUNISTS WAIT TILL NEXT YEAR SCARCELY HAD THE SINO-SOVIET TALKS GOTTEN UNDERWAY THAN THE MEETING HEADED FOR COLLAPSE . IT DID NOT MUCH MATTER WHEN RED CHINA'S SEVEN-MAN DELEGATION WOULD PACK THEIR BAGS AND ACTUALLY LEAVE MOSCOW ; BACK HOME PEKING'S PEOPLE'S DAILY SEEMED READY TO CALL IT QUITS . " WE WANT UNITY, NOT A SPLIT, " SAID THE VOICE OF…

Labels: **peking,red,soviet**

Score: 28

Document Name: TIME302

Extract from the document: …KENYA THE RETURN OF BURNING SPEAR IN DAZZLING SUNLIGHT LAST WEEK, 30,000 SINGING, DANCING AFRICANS GATHERED BEFORE NAIROBI'S MINISTRY OF WORKS . A GREAT ROAR WENT UP AS TWO SOLEMN MEN EMERGED . ONE WAS KENYA'S BRITISH GOVERNOR MALCOLM MACDONALD . THE OTHER, WEARING HIS CUSTOMARY LEATHER JACKET AND BEADED BEANIE, WAS BURLY JOMO …

Labels: **kenyatta,kenya**

Score: 25

Document Name: TIME400

Extract from the document: …GREAT BRITAIN THE SAGA OF POLISH PETER LIKE THE OVERTURNING OF A DEEPLY EMBEDDED ROCK, THE PROFUMO SCANDAL CAUSED A FRANTIC SCURRYING OF A GREAT MANY ODD HUMAN INSECTS . ONE OF THE CRAWLIEST FIGURES TO EMERGE WAS THAT OF PETER RACHMAN, WHO MAY, OR MAY NOT, BE DEAD . LAST WEEK PRESS AND PARLIAMENT WERE ABUZZ WITH HIS SORDID STORY . RACHMAN LOOKED …

Labels: **rachman**

Score: 21

Document Name: TIME391

Extract from the document: … SOUTH AFRICA FAMILY TROUBLES FAMILY DAY IN SOUTH

AFRICA IS AN EXPANDED VERSION OF MOTHER'S OR FATHER'S DAY A TIME FOR ALL KINFOLK TO GET TOGETHER . SOUTH AFRICA'S WHITES AND BLACKS LAST WEEK CELEBRATED THE HOLIDAY IN IRONICALLY CONTRASTING WAYS . WHILE WHITES PICNICKED OR FROLICKED ON BEACHES, THOUSANDS OF BLACKS MOURNED THE ABSENCE OF …

Labels: **south**

Score: 13

Document Name: TIME149

Extract from the document: **…**EAST AFRICA THE ASIANS IN THEIR MIDST FOR MANY EUROPEAN SETTLERS, AFRICA FOR THE AFRICANS " SIMPLY MEANS PACKING UP AND GOING HOME, PAINFUL THOUGH IT MAY BE . THE FUTURE IS FAR DARKER FOR THE ASIANS IN EAST AFRICA, WHO HAVE LONG FORMED A PRECARIOUS MIDDLE CLASS . DESPISED BY COLOR-CONSCIOUS WHITES, …

Labels: **African**

Score: 10

Document Name: TIME068

Extract from the document: …EGYPT SURPRISE AT SUEZ WHEN EGYPT'S PRESIDENT GAMAL ABDEL NASSER GRABBED THE SUEZ CANAL 6F YEARS AGO, HIS BITTER ENEMIES IN EUROPE PREDICTED THAT THE BIG DITCH WOULD SOON BE FILLED WITH SILT AND THAT UNTRAINED EGYPTIAN PILOTS WOULD NEVER BE ABLE TO STEER SHIPPING THROUGH SAFELY . THE CRITICS TURNED OUT TO BE WRONG ON BOTH COUNTS . EGYPT HAS …

Labels: **egypt**

Score: 4

Document Name: TIME444

Extract from the document: …AMEN ! FOR GENERATIONS THE WHITE WOMEN'S BURDEN IN STEAMY SOUTHEAST ASIA HAS BEEN SHOULDERED BY AMAHS, THE SOFTFOOTED, TOUGH-FIBERED MAIDSERVANTS WHO WERE RECRUITED FROM THE CHINESE MAINLAND . WHILE THE AMAH (LITERALLY, " LITTLE MOTHER /) COOKED, CLEANED AND LOOKED AFTER THE CHILDREN, THE COLONEL'S LADY…

Labels: **time,white,malayan**

Score: 2

5. Conclusions and Future Work

Firstly, the experiments confirm that paritional clustering of normalized DF vectors does reveal the semantic relationship and groups the semantically coherent words together. Secondly, the experiments testify our idea that words around the cluster center can be used as representative words. Thirdly, the collection of representative words of various clusters, seem to be abstract enough to label documents. This is particularly interesting since, in the course of our experiments, we use a maximum of 500 words (5 each from 100 clusters) for labeling documents out of the entire dictionary which comprises 20000 words. Thus, we have been successful in reducing the dictionary by 1/40th without any significant loss of words that could be used as a label (as our experiments show). Lastly, our experiments validate the utility of term frequencies as a meaningful and simple statistic in assigning clusters to documents and thus assigning labels to documents.

We have used partitional clustering techniques in the course of our experiments. We would like to use soft clustering techniques where a word can be assigned to more than one cluster to extend this work. As a motivating example (from our experiments) towards the same, "south" is related to Vietnam (as a lot of cold war events are centered around south Vietnam) and to "Africa" (south Africa as a country features in the corpus, although very rarely). We find that "south" has been clustered with "Vietnam" in the same cluster. We would like to devise soft clustering techniques which make use of co-occurence frequencies so that even the slightest semantic relationships (such as that between "south" and "Africa" which occur together very rarely) could be made explicit. We would also like to work with clustering of n-grams rather than single words. Although, it would obviously be more computationally expensive compared to the word clustering approaches, phrases such as "south Africa" and "cold war" would arguably have much more descriptive power compared to sets of words.

References

1. McCallum, A., and Nigam. K. (1999) “Text classification by bootstrapping with keywords, EM and shrinkage.” ACL Workshop on Unsupervised Learning in Natural Language Processing.

2. McQueen, J.B. 1967, Some Methods of Classification and Analysis of Multivariate Observations, 1967

3. Lewis, D., and Gale, W. (1994). A sequential algorithm for training text classifiers. SIGIR-94.

4. McCallum, A., and Nigam, K. (1998a). A comparison of event models for naïve Bayes text classification. AAAI-98 Workshop on Learning for Text Categorization.

5. Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. ECML-98.

6. Lang, 1995. NewsWeeder: Learning to Filter Netnews, ICML 1995

7. Koller & Sahami, 1997, Hierarchically Classifying Documents Using Very Few Words, ICML - 1997

8. Nigam, K., McCallum, A., Thrun, S., and Mitchell, T. (2000). Text classification from labeled and unlabeled documents using EM. Machine Learning, 39.

9. Blum, A., and Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. COLT-98.

10. Bockhorst, J., and Craven, M. (2002). Exploiting relations among concepts to acquire weakly labeled training data. ICML-02.

11. Ghani, R. (2002). Combining labeled and unlabeled data for multiclass text categorization. ICML-2002.

12. Goldman, S. and Zhou, Y. (2000). Enhancing supervised learning with unlabeled data. ICML-2000.

13. Denis, F. (1998). PAC learning from positive statistical queries. ALT-1998.

14. Liu, B., Lee, W. S., Yu, P., and Li, X. (2002). Partially supervised classification of text documents. ICML-02.

15. Yu, H., Han, J., Chang, K. (2002). PEBL: Positive example based learning for Web page classification using SVM. KDD-02.

16. Bing Liu et al, 2004, Text Classification by Labeling Words, AAAI-2004

17. Youngjoong & Jungyan, 2000, Automatic Text Categorization by Unsupervised Learning, COLING 2000

18. Sabine Bergler. “Collocation patterns for verbs of reported speech--a corpus analysis of time Magazine corpus”. Technical: report, Brandeis University Computer Science,. 1990.

Chapter 4

Unsupervised Learning from URL Corpora

1. Introduction

Techniques for unsupervised learning from web documents (and hypertext) have gained a lot of attention over the past few years. Information sources for such techniques usually include hyperlink structure, the text of the web document, the latent structure in the markup language used (such as differential weighting for titles, table headings etc. in HTML), anchor texts of links to and from the web document etc. Most of the techniques proposed have ignored the URL information. Here we attempt the novel problem of unsupervised learning from corpora of URLs. Firstly, we present a similarity metric for URL pairs, which is very different from its counterparts for document (text or hypertext) pairs. Secondly, we attempt Hierarchical agglomerative clustering based on the similarity metric. Thirdly, we go on to show that the measure is useful for keyword identification from topical URL corpora. Lastly, we attempt keyword identification on heterogeneous URL corpora. Given that URLs are small entities, our techniques are magnitudes faster than unsupervised techniques on full-text corpora and require far less information than the latter. To the best of our knowledge, this is the first attempt on unsupervised learning from URL information.

2. Related Work

Techniques for web document clustering are very mature in the search engine context, where clustering of results for a query has been widely and successfully experimented ([1], [2], [3], [4], [5], [6], [7]). Vivisimo[2], iBoogie[3] and Clusty[4] are search engines that do clustering of search results. Although clustering of pure web document corpora (in contrast to web search result corpora) have received far less attention, an

[2] http://www.vivisimo.com
[3] http://www.iboogie.com
[4] http://www.clusty.com

attempt at finding the most important features for web document clustering using an evolutionary algorithm [8] is notable in this regard. A large benchmark dataset [9] has been published to aid clustering tasks on web corpora, although the same has been used more for classification ([10], [11]) than clustering related tasks [12]. This follows the increased attention that web document classification has traditionally been getting, compared to its unsupervised counterpart ([13], [14]). The first attempt on harnessing URL based info for a machine learning task is the MeURLin[5] system, a URL based Web Page Classifier, details of which can be found elsewhere ([15], [16]). Our work involves unsupervised clustering of URL corpora, and is significantly different from MeURLin, in that the latter is a supervised task, doesn't involve a pair wise URL similarity measure and that the tokenization of URLs in the latter is biased by the training set used.

Topic Detection and Tracking [17] is an emerging field in text mining. Both the sub-fields aim to annotate a document with the topic that it refers to; the former utilizes a closed set of topics (or labels), whereas the latter can use a new topic label for an incoming document. We attempt to summarize homogenous corpora of URLs by sets of keywords (or keyword fragments), and then apply the approach to heterogeneous corpora to evaluate the performance. Although summarization of URL corpora is a novel problem, multi-document summarization has been gaining increasing significance over the past many years. Generating a ranked list of descriptive keywords from homogenous web-document corpora using link and content based information is a related attempted [22] problem. Related work in multi-document summarization largely focuses on generating sets or sequences of sentences rather than sets of keywords for summaries. Summarization of a cluster of documents using the centroid of the cluster [18] has been attempted successfully and is one of the earliest works in this area. WebInEssence[6] and MEAD[7] are tools of interest in this field. Usage of Katz's K-mixture model [19] for sentence ranking in multi-document summarization has been investigated. Comparisons of document clustering and redundancy reduction techniques for multi-document summarization [21] and attempts at developing infrastructure for evaluation of multi-

[5] http://wing.comp.nus.edu.sg/meurlin/
[6] http://tangra.si.umich.edu/clair/home/web.html
[7] http://tangra.si.umich.edu/clair/home/mead.html

document summaries [20] are other interesting works in the context of the problem at hand.

3. URLs as an Information Source

3.1 Information Content of a Single URL: The Use of Background Knowledge

We would like to state upfront that the discussion in this subsection is to enable to appreciate the information content of a single URL, given some background knowledge. Although it is more relevant for a supervised task which organizes and uses background knowledge, we include this to emphasize that URLs are an important information source and thereby justify our intention of mining URL corpora. Consider the URL www.cs.cmu.edu . Given the background knowledge that edu refers to educational institutions and cs refers to computer science, we could readily infer that it is the homepage of the CS department/school in a university which bears the acronym CMU. This is illustrative about the info that a URL (coupled with the background knowledge) can hold, and is suggestive of the effectiveness of supervised learning techniques on URLs. This motivates the usage of learning from URLs.

3.2 URL Corpora: Motivation for Unsupervised Learning

Unsupervised learning usually involves learning from a collection of entities; we choose to delve into the possibilities of unsupervised learning from URL corpora. The set of URLs <www.iisc.ernet.in , www.iitkgp.ernet.in , www.iitm.ernet.in > all belong to research/educational institutions in India and ernet is a suffix for the Indian “Education and Research Network”. The occurrence of ernet can enable the above URLs to cluster together due to the common factor that they belong to the same class of institutions. Further, www.abc.ernet.in can be inferred to be somehow related to the above URLs due to the common suffix. For a cluster of URLs that have the ernet prefix, “ernet” could be a descriptive keyword. URLs are a scanty resource for mining and hence mining just the

URL words (delimited) would not possibly suffice. Such mining would not recognize the similarity between www.kerala.gov.in and www.newkerala.com (in that both relate to the same state called Kerala in South India). This motivates the usage of largest common substring based similarity measurements for clustering. Although very hostile cases, such as www.whitehouse.gov (US Government Site) and www.whitehorse.com (A website design company), would surely plague the similarity measure, our experiments show that such cases aren't frequent enough to cause too much of a harm.

3.3 Variable Information Content of URL Segments

URLs are variably delimited (as opposed to all delimiters carrying the same meaning or significance) sequence (as opposed to sets) of words. We refer to a URL segment as a delimited segment of text in the URL, with every non-alphanumeric character treated as a delimiter (with exceptions for ASCII characters, such as %20 representing whitespace). Consider the URL: http://www.cs.abc.edu/courses/current/cs511/assignments , which is a hypothetical example for the assignments webpage of a course CS511 offered in the current semester by the CS department of the ABC University. 'Current' can be made sense of, in the context of 'courses' only. Similarly, 'CS511' makes sense only in the context of 'courses'. The vice versa isn't always true. For instance, 'courses' make sense even in the absence of 'CS511' and 'assignments'. We argue that 'courses' is a better determiner than 'current' and 'cs511' and conveys more information. This translates to an interesting conclusion; that information content decreases as we go down the URL. The first sequence of URL segments, the one starting right after the protocol specifier and going on till the next non-dot delimiter, commonly referred to as a hostname, notably presents an exception with regard to information content breakup between segments. In hostname www.cs.abc.edu , CS makes sense only in the context of 'abc' whereas 'abc' makes sense only in the context of edu. It can be readily inferred that the vice versa isn't true. We sum up our conclusions in this regard to say that information content of a segment decreases as we go down the URL, whereas it is exactly the opposite in the case of the hostname.

4. URL-Sim: A Similarity Measure for URL Pairs

We present a similarity measure for URL pairs which we consistently use through the experiments in the rest of the paper. The similarity computation is done by means of 4 simple phases, each of which is described in a dedicated subsection therein. The similarity computation takes in 2-tuples, <URL1, URL2> and computes the similarity as a single numeric value.

4.1. Pre-Processing

We remove the scheme/protocol tag from each of the URLs, URL1 and URL2. This is based on the intuition that clusters can rarely be categorized by the scheme information and that scheme information is very rarely a determiner for that category that a URL falls in. Further, we remove stopwords from the URLs. Stopwords are words with little determining power, and their removal is often an implicit pre-processing step in most information retrieval or text mining tasks. We have currently identified the set of stopwords as {com, net}. Although both of them were intended to reveal some detail about the page, their usage in the current scenario is a firm motivating factor for labeling them as stopwords. Further, as the last part of phase 1, the order of URL segments in the hostname are reversed, so that the URL segments in the output from phase 1 would be in the decreasing order of information content, as per the intuition outlined in Section 3.3. Thus, at the end of phase 1, we get a newer trimmed URL from the original input URL. Some examples are cited below.

Table 1. Phase 1 Tasks depicted as Input-Output Pairs

Input to Phase 1	Output from Phase 1
http://www.iitm.ac.in/students	in.ac.iitm/students
http://www.cs.cmu.edu/afs	edu.cmu.cs/afs

4.2. Tokenizing and Weight Tagging

Tokenizing involves splitting up the URLs into segments. We refer to a URL segment as a delimited segment of text in the URL, with every non-alphanumeric

character treated as a delimiter (with exceptions for ASCII characters, such as %20 representing whitespace). Each segment has an associated level, which stands for the number of '/' occurring before it in the pre-processed URL, and an associated rank, which is the sequential order of the segment in the pre-processed URL among the segments which are in its level. We have a hard-coded function, total_weight(level), which (as given below) gives the total weight that a level is assigned with. Total_weight(level) gives the sum of the weights of the segments in that level.

Table 2. Hard-coded Weight Assignments to URL Levels

Level	Total_Weight(Level)
0	10
1	8
2	6
3	4
4	2
5 – upwards	0

The split-up of weights between the segments in a level is dependent on the rank of each segment and the total number of segments in the level. The function to determine the weight of a segment (the total number of segments in a level represented by total_segs(level)) is given as below.

Weight(Segment s) = (total_segs(s.level()) + 1 – s.rank())* Total_Weight(s.level()) /(1+2+…+total_segs(s.level())	(1)

The expression is quite simple in that, it distributes weights in a level in the reverse order of ranks. Thus, if there are two tokens in a level, with 'a' bearing rank 1 and 'b' bearing rank 2, the ratio of weights of 'a' and 'b' would be 2:1, with the added constraint that the total weight would add-up to the total weight allocated to the level. Thus, at the end of this level, each segment of the URL would be tagged with a weight. An example is included below.

Table 3. Weight Breakups for a Sample URL

Pre-processed URL	Segment	Weight
in.ac.iitm/students	in	5.00
	ac	3.33
	iitm	1.67
	students	8.00

4.3. Similarity Computation

The final similarity computation for URL pairs, involves pair wise matching of each segment from URL1 with each segment from URL2. The similarity value is initialized to zero, with each segment pair adding a value to the similarity depending on the length of the segments, weights associated with the segments, and the length of the largest common substring. The function is briefly summarized in the table. The increment is computed as the weighted average of the 'amount' of match between the segments. If both the segments are the same (strings), increment would be the average of their weights.

Table 4. Similarity Computation for URL Pairs (in Pseudocode)

Similarity Computation for URL Pairs

URL-Sim(URL URL1, URL URL2)
{
URL-Sim-Val = 0;
for each pair <seg1, seg2>, seg1 Є URL1 and seg2 Є URL2
{
Let C = Length of the Largest Common Substring, str,
between seg1 and seg2;
Increment = ((seg1.weight()*C/seg1.length()) +
(seg2.weight()*C/seg2.length()))/2.0;
URL-Sim-Val = URL-Sim-Val + Increment;
}
return URL-Sim-Val;
}

5. Unsupervised Learning Using URL-Sim

5.1 Clustering

Having obtained the pair wise similarity measures for every URL pair in the URL corpus, we apply the overly popular hierarchical agglomerative clustering algorithm [23] on the corpus. It starts of with as many singleton clusters as there are URLs and goes on to merge two closest clusters per iteration. The similarity between clusters are taken as the average of the similarity between URL pairs <URL1, URL2> the first item in the pair from one cluster and the second item from the other. The average purity of the clusters is determined using the formula below.

Purity(Clustering C) = ($\sum_{Ci \in C}$ {Cardinality of the most frequent label in Ci}) / |Total elements|	(2)

5.2 Keyword Detection

The second unsupervised learning task that we attempt is that of keyword detection on a homogeneous corpus. The algorithm is summarized in the pseudocode below. This algorithm reuses most of the URL-Sim algorithm. It outputs a scored list of words for the input corpus. As we have no means of evaluating the performance by means of a numerical measure, we output a few high-scoring strings for experiments on keyword identification. Also note that, the strings output may not be full-keywords, but may just be keyword fragments, as we choose to score the largest common substrings rather than segments themselves.

Table 5. Keyword Detection Algorithm (in Pseudocode)

Keyword Detection Algorithm
Keyword_Detection(Corpus C) { for every possible string s, s.score = 0;

```
 for every pair <URL1, URL2>, URL1, URL2 Є C
 {
  Pre-process URL1 and URL2 (Section 4.1) and weight-tag
   their segments (Section 4.2);
  for each pair <seg1, seg2>, seg1 Є URL1 and seg2 Є URL2
  {
    Let C = Length of the Largest Common Substring, str,
     between seg1 and seg2;
    Increment = ((seg1.weight()*C/seg1.length()) +
     (seg2.weight()*C/seg2.length()))/2.0;
    str.score = str.score + Increment;
   }
 }
 Output the list of strings in the descending order of scores
}
```

6. Experiments

6.1. Corpora Used

For our experiments, we use subsets of the standard corpora such as the WebKB[8] (4 university) dataset, BankSearch[9] Dataset [9] (the subset used is detailed with the results) and some corpora which we collected from Google[10]. We decided to collect our own datasets rather than fully relying on standard corpora because of the fact that the standard corpora were intended to be used as document clustering datasets, and hence good/bad performance of our techniques on them may/may not reflect on the quality of our techniques. For instance, our clustering techniques work very well for the WebKB dataset, because the URLs themselves contain the category labels (cornell.edu is part of every URL categorized under Cornell). A motivating example would be to say that www.india.gov.in, can at best be clustered into a category of Indian or Governmental websites, but never into a cluster of reports on the Indian Prime Minister even though the

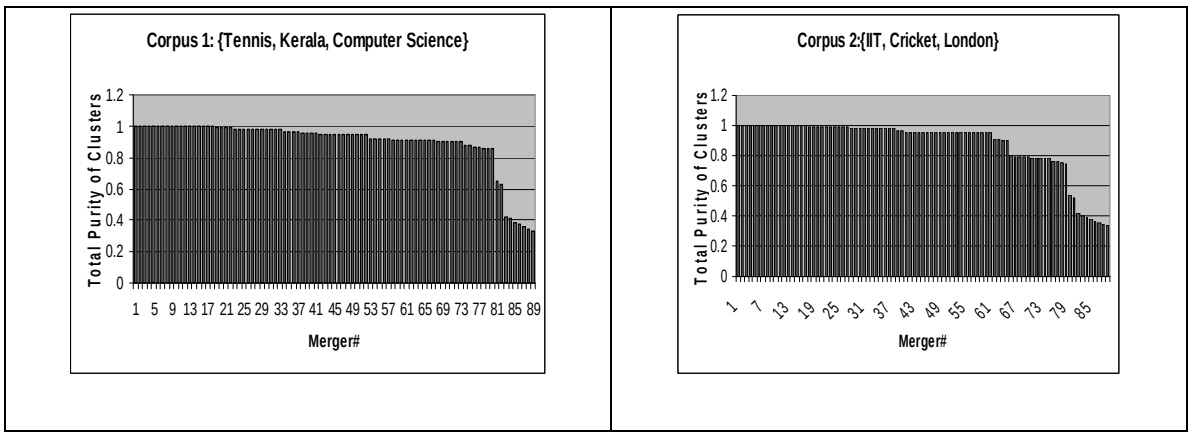
[8] http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/
[9] http://www.banksearchdataset.info
[10] http://www.google.com

current version of the webpage may warrant its inclusion into such a cluster. Contents may change, but URLs don't (justifying their usage as URLs). For gathering corpora by means of Google, we used a query Q and obtained the top 30 results for that query from Google. Many such queries were issued, and result sets from various queries were merged to form URL corpora, each entry labeled with the query for which it was obtained as a result. Hereafter, a set labeled with a term or phrase without reference to a standard dataset, refers to the set of first 30 result pages obtained for it from Google (as in Kerala, Computer Science etc.).

6.2. Clustering Results

We present a sequence of charts herewith summarizing the results of our clustering experiments on various corpora. For a cluster having 30 documents each of 3 different labels, the final clustering purity and initial clustering purity will always be 0.33 (=30/90) and 1.0 (all singleton clusters) respectively. We consistently use corpora containing exactly three labels for our experiments to aid visual comparison of result graphs. The quality can (intuitively) be assessed by means of how late the curve declines (or starts to decline). The later and the sharper the decline of the purity curve, the better the clustering (and hence the similarity measure used for clustering). For ease of evaluation, we provide the number of 'mistakes' by the clustering algorithm in the first 30% and 50% of the mergers. A 'mistake' is defined as a merger that results in a decrease of total purity.

Table 6. Purity Plots for Various Datasets Used in the Experiments



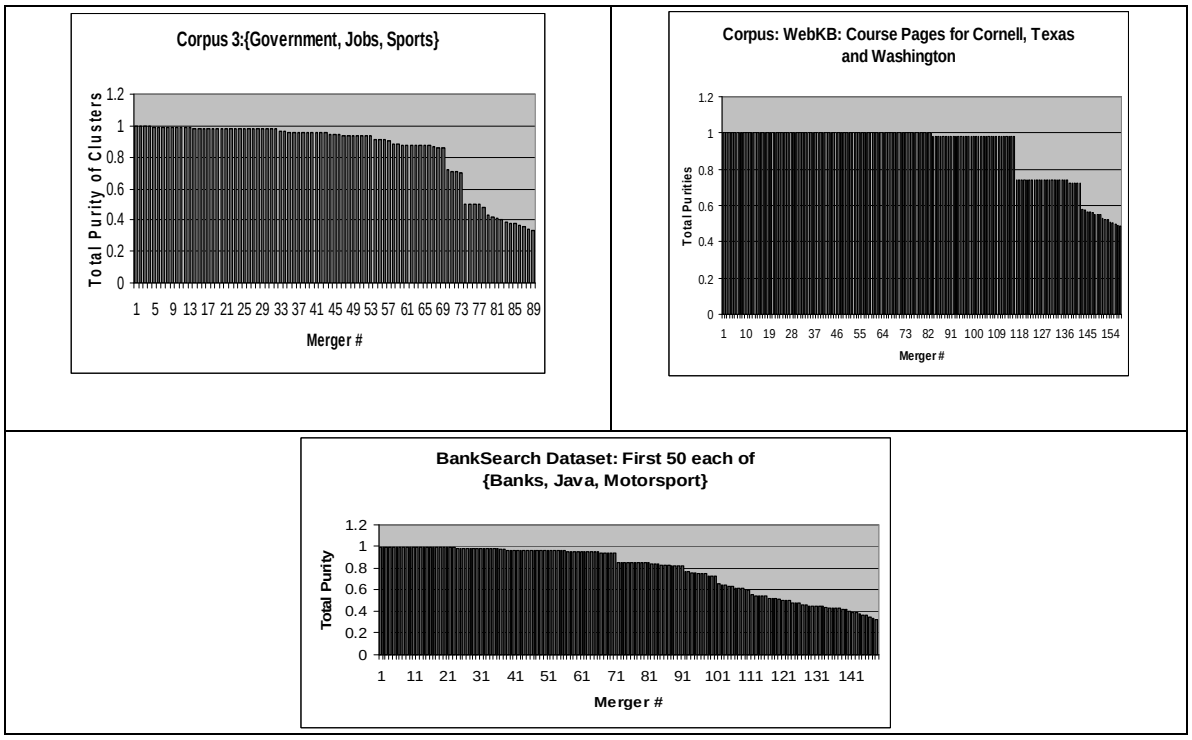


Table 7. Log of "Mistakes" Made by the Clustering Algorithm

Corpus	**Count (No. of documents in the corpus)**	**Mistakes in first 30% mergers**	**Mistakes in first 50% mergers**
Corpus 1:{Tennis, Kerala, Computer Science}	90	1	3
Corpus 2:{IIT, Cricket, London}	90	1	3
Corpus 3: {Government, Jobs, Sports}	90	2	5
Corpus 4: WebKB {Cornell, Texas, Washington} course pages	159	0	0
Corpus 5: BankSearch, first 50 pages each of {Banks, Java, MotorSport}	150	5	10

6.3. Keyword Identification Results

We present the results (word, score tuples in descending order of scores; as many of the top n-tuples as we deem to be relevant) of the keyword identification experiments, both for homogenous corpuses, as well as for each of the heterogeneous corpuses used for the clustering experiments.

Table 8. Results of the Keyword Identification Algorithm on Homogenous corpora

Homogeneous Corpus	List of Sorted Keyword Score Tuples
Cricket	<**cricket**, 1161> <co, 243>
Computer Science	<**cs**,1094> <edu,313>
Government	<**gov**,1007> <go,106>
IIT	<**iit**,1292> <in,125>
Jobs	<**jobs**,706> <co,86>
Kerala	<**kerala**,1003> <keral,114>
London	<uk,300> <**london**,192>
Sports	<**sports**,402> <sport,224>
Tennis	<**tennis**,748> <en,230>
Cornell (WebKB)	<info, 6560> <cs,5222> <courses, 3780> <**cornell**, 3095>
Texas (WebKB)	<cs,7263> <users,5624> <**utexas**,2343> <edu,1171>
Washington (WebKB)	<education,22800> <edu,17555> <courses,17100> <cs,16026> <**washington**,9880>
Banks (BankSearch)	<co,3057> <uk,1258> <**bank**,677>
Java (BankSearch)	<**java**,2443> <ava,602> <ja,245>
MotorSport (BankSearch)	<in,473> <**motorsport**,342> <or,324> <race,285> <sport,246>

Table 9. Results of the Keyword Identification Algorithm on Heterogenous corpora

Heterogeneous Corpus	Keywords
Corpus 1:{Tennis, Kerala, Computer Science}	<**cs**,1321> <**kerala**,1003> <**tennis**,748> <co,575> <en,507>
Corpus 2:{IIT, Cricket, London}	<**iit**,1292> <**cricket**,1161> <co,957> <uk,459> <ac,390>
Corpus 3: {Government, Jobs, Sports}	<**gov**,1348> <**jobs**,772> <**sports**,402> <co,365>
Corpus 4: WebKB {Cornell, Texas, Washington} course pages	<cs,110190> <education, 22800> <courses,17864> <**washington**,9753>
Corpus 5: BankSearch, first 50 pages each of {Banks, Java, MotorSport}	<co,5183> <in,3910> <**java**,2443> <al,2042> <es,2031>

7. Conclusions, Contributions and Future Work

By way of our observations from the experiments, we deem ourselves competent enough to assert that URL-Sim performs very well as a similarity measure for URL pairs. The number of mistakes that HAC makes (with the URL-Sim measure) is very minimal among the first few mergers. Although URL-Sim doesn't take the content of the target web-pages into account (which is by far, the major concern for text clustering tasks), we could use HAC with URL-Sim and perform the first, say 30%, of the mergers and arrive at initial clusters which would provide a good deal of insight into the kind of clusters present in the corpus. The keyword identification experiments also have performed exceedingly well on homogeneous corpora with the main topic-word being ranked as the highest scoring word in nine out of the 15 corpora that we chose to test with. As can be seen from the results, the techniques worked unexpectedly fine even for heterogeneous corpora. At this juncture, we briefly summarize our contributions by way of this paper. Firstly, this is the first attempt (to the best of our knowledge) on unsupervised learning from URL corpora. Secondly, we lay down a similarity measure for URL pairs, URL-Sim, which makes use of the intuitive structure and differential information content of the URL. Thirdly, we demonstrate the feasibility and accuracy of clustering of URL corpora based on the URL-Sim measure. Fourthly, we present an approach utilizing the URL-Sim measure for keyword extraction from homogeneous URL corpora. Lastly, we show that the keyword extraction algorithm works well even for heterogeneous URL corpora.

Future work in this regard could address the application of further unsupervised learning techniques such as association rule mining on URL corpora. Usage of ontologies to enhance the URL-Sim function could be explored. Techniques to project the URLs into a vector space could be useful as it would aid usage of partitional clustering algorithms such as K-Means.

References

1. Zamir, Etzioni, Madani, Karp, "Fast and Intuitive Clustering of Web Documents", Proc of KDD-1997

2. Zamir & Etzioni, “Web Document Clustering: A Feasibility Demonstration”, Proceedings of ACM SIGIR 1998, pp. 46-54

3. Zamir, Etzioni, Madanim, “Grouper: A dynamic clustering interface to web search results”, Proceedings of the 8th WWW Conference, 1999

4. He, Zha, Ding, Simon, “Web document clustering using hyperlink structures”, Computational Statistics and Data Analysis, 2002

5. Strehl, Ghosh and Mooney, “Impact of Similarity Measures on Web-Page Clustering”, Proceedings of the AAAI-2000 Workshop on Artificial Intelligence for Web Search, 2000

6. Deepak, John and Parameswaran, “Context Disambiguation in Web Search Results”, Proceedings of the IEEE Intl. Conference on Web Services (ICWS-2004), 2004

7. Wang, Kitsuregawa, “Link-based Clustering of Web Search Results”, Proceedings of the WAIM-2001, SpringerLink, 2001

8. Sinka, Corne, Building, “Evolving Document Features for Web Document Clustering: A Feasibility Study”, Proceedings of the IEEE Congress on Evolutionary Computation, 2004

9. Sinka, Corne, “The BankSearch web document dataset: investigating unsupervised clustering and category similarity”, Journal of Network and Computer Applications, 2005

10. Benbrahim, Bramer, “An empirical study for hypertext categorization”, IEEE International Conference on Systems, Man and Cybernetics, 2004, pp.5952-5957

11. M. Kovacevic, M. Diligenti, M. Gori, V. Milutinovic. “Visual Adjacency Multigraphs - a Novel Approach for a Web Page Classification.” Proceedings of the ECML/PKDD Workshop on Statistical Approaches to Web Mining, 2004

12. Mark P. Sinka, and David W. Corne, “Measuring Effectiveness of Text-Decorated HTML Tags in Web Document Clustering”, in Proceedings of the IADIS International WWW/Internet 2004 Conference, Madrid, Spain, 2004.

13. Tsukada, Washio, Motoda, “Automatic Web-Page Classification by Using Machine Learning Methods”, Web Intelligence, 2001

14. Yu, Hang, Chang, “PEBL: Web page classification without negative examples”, IEEE Transactions on Knowledge and Data Engineering, 2004

15. Kan, Thi, “Fast webpage classification using URL Features”, Proceedings of the Conference on Information and Knowledge Management, CIKM-2005, 2005
16. Kan, “Web Page Classification without the Web Page”, Proc. of the 13th WWW Conference, 2004
17. Allan, “Introduction to Topic Detection and Tracking”, The Kluwer Intl. Series on Information Retrieval, 2002
18. Radev, Jing, Sys, Tam, “Centroid-based summarization of multiple documents”, Information Processing and Management: An International Journal, Elsevier, 2004
19. Saravanan, Raman, Ravindran, “A Probabilistic Approach to Multi-Document Summarization for Generating a Tiled Summary”, Proc of the ICCIMA-2005, 2005
20. Saggion, Radev, Teufel, Lam and Strassel, “Developing infrastructure for the evaluation of Single and Multi-Document Summarization systems in a cross-lingual environment”, Proceedings of LREC-2002, 2002
21. Seki, Eguchi, Kando, “User-Focussed Multi-Document Summarization with Paragraph Clustering and Sentence-Type Filtering”, Working Notes of NTCIR-4, 2004
22. Deepak & John, “Identifying the subject of small, sparsely linked collections from a web community”, International Journal on Web Based Communities, Inderscience, 2004
23. Willet, “Recent Trends in Hierarchical Document Clustering: A Critical Review”, Information Processing and Management, 1988

Chapter 5

Differential Voting in Case Based Spam Filtering

1. Introduction

Case-based reasoning has been shown to be of considerable value in a spam filtering task ([2] and [3]). Case base in a CBR system incorporates all the knowledge of the system. Given that case base consists of cases (which are traditionally represented in the vector-space model), we see considerable potential for usage of data mining tasks to improve the CBR system, esp. in the case of a classification task as spam filtering. As far as we know, there have been no attempts to make use of the skewed distribution (hereafter referred to as patterns) in the case base in a CBR system. We propose that each case in the case base be associated with a voting power, which incorporates the knowledge of the neighborhood of the case, which can then be made use of to classify a test case. We propose various algorithms for computing the voting power of a case, and go on to show that many of them work exceedingly well in comparison with the traditional techniques. Further, the complexity of re-computing the voting powers when an addition or deletion occurs to the case base is linear in the number of cases in the case base. To the best of our knowledge, this is the first study to incorporate some basic data mining techniques within in a CBR system.

2. Related Work

It has been shown [1] that memory based approaches for spam filtering work significantly better than well studied naïve Bayesian approaches. Further, they go on to say that it might probably be because of the fact that there are many more types of messages rather than just spam and legitimate. A more recent work [2] proposes new methods of feature selection based on spam and non-spam vocabularies and asserts that a CBR approach to spam filtering can effectively track and adapt to the changing behavior of spammers and legitimate mails (concept drift) and provides methods for the same. It uses the conventional and intuitive CBR approach of majority voting (the voters being

the neighbors of the test case in the case base) to flag a message as either spam or legitimate. Another work [3] incorporates various ideas specific to spam filtering. Firstly, it incorporates the notion that a legitimate message flagged as spam is much costlier than a spam message labeled legitimate. Further, it presents a significant departure from the conventional CBR model in that it incorporates differential weighting of votes, viz., the closer a neighbor is, to the test case, the higher would be the value of it's vote (we call this technique Diff-CBR hereafter in this paper). It also incorporates differential weighting of features in the vector space. In this paper, we compare our techniques to the approaches used in the latter two papers.

3. Motivation and Justification

In the day-to-day life, an average web user comes across a wide range of spam and legitimate mails. Intuitively, most mails fall into more categories than just two classes as spam and legitimate. Many of the spam mails that the authors receive fall into categories such as "interest free home loans", "mortgage", "easy university degree" and lastly, the very popular class of porn spam mails. We assert, with further clarification of the assertion in a later section, that there are clusters of spam mails, i.e., there are definite patterns or clusters in the case-base of the CBR Spam Filter. We suspect that making use of such patterns would improve the performance of a CBR spam filter considerably. In the following subsections, we describe the dataset used and justify our assumption that clusters exist in mail corpuses by experiments on the corpus.

We choose to use the SpamBase Database (hereafter referred to as the corpus) compiled by Goerge Forman of HP Labs. It is available through the University of California Irvine Machine Learning Repository[11]. SpamBase is a collection of 4601 messages, each message represented as a labeled (as either spam or legitimate) vector of 57 selected features and contains 39.4% spam messages.

[11] http://www.ics.uci.edu/~mlearn/MLRepository.html

In order to justify our claim that there are clusters in mail corpuses, we proceed to show that there are clusters in the corpus and that they are pure enough to be made use of. We applied the traditional K-means clustering algorithm with K = 15 with varying initial cluster centers on the corpus. The presence of clusters would reveal a skewed distribution of messages among clusters and that is exactly what we ran into. Among the 15 clusters that we obtained, one of them had 42% of the corpus and only 5 clusters had more than 5% of the corpus. The percentages of the corpus in each of the 15 clusters are listed in table 1.

Table 1. Distribution in K-Means Clusters with K = 15

Percentages of the Corpus in Clusters														
11	1	20	3	6	4	3	1	1	2	42	1	1	0	6

Having justified our claim that there are clusters in the corpus, we go on to show that the clusters are pure enough (using the labels in the corpus). The overall purity of a clustering is the weighted average of the purities of the different clusters. The following is a plot of the weighted average of the purity of the K-means clusters against k. As can be seen, the purity plot meanders around 0.78, which shows that the clusters are pure enough.

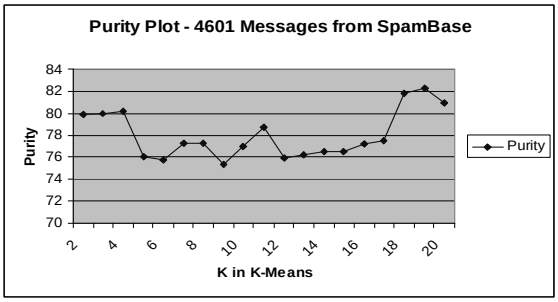


Figure 1. Total purity of Clusters with varying K

Thus, there are clusters in the corpus which are pure enough to be used to be exploited in a CBR system. Further, we go on to propose various techniques to make use of the skewed distribution in the CBR framework and compare it with the state-of-the-art techniques such as simple CBR and diff-CBR.

4. Broad Methodology

In each of the techniques that we propose (in sections that follow), we use the concept of voting powers for a case in a static case base. Given a case base, we can associate each case with a voting power depending on the cases in its vicinity and their labels. In order to compute voting power, we can either consider

- The k-nearest neighbors (k-NN) of the case (and their labels) in the case base
- All cases which are no farther from the case than a given distance (which can be taken as an input parameter)

We use the k-NN of a test case consistently through this paper. Given a test case and the ultimate aim of classifying it as either spam or legitimate, we need two functions:

- Confidence_Spam(Test_Case T, k-NN of the test case in the Case Base) which returns the confidence of the test case being spam and
- Confidence_Legitimate(Test_Case T, k-NN of the test case in the Case Base) which returns the confidence of the test case being legitimate

The intuitive algorithm for classification requires just a ternary operator in most programming languages and is:

- Flag(Test_Case T) = Confidence_Spam(T,k-NN) > Confidence_Legitimate(T,k-NN) ? “Legitimate” : ”Spam”

In cases where the confidences are equal, we argue that we should “play safe” and label it as legitimate.

The Confidence functions that we propose make use of the voting powers of each of the k-NN neighbors, and optionally their distances from the test case. Both the confidence functions use the same algorithm, except for the fact that one considers only spam elements in the k-NN and vice versa. We present a broad framework of the classifier algorithm.

Table 2. Confidence Calculation, X∈{spam, legitimate}

Confidence Calculation
Confidence_X(Test_Case T, k-NN of T in the Case Base){

```
    confidence = 0.0;
    for each C among the k-NN neighbors {
        if(label(C) == X){
            add_pwr = Voting_Power(C);
            optionally,
                    add_pwr = add_pwr / distance(Test Case, C);
             confidence = confidence + additive_power; }
    }
    return confidence;
}
```

Having introduced as many primitives as has been done, each algorithm can be specified by the voting power computation function and as to whether it involves the optional step in the algorithm. We call algorithms that include the optional step as *distance-weighting algorithms* for the sake of brevity hereafter in this paper.

5. Techniques for Spam Filtering Using CBR

We present two techniques that have already appeared in literature followed by 6 techniques that we propose, to improve the performance of the CBR Spam Filter. All the descriptions use the primitives introduced in the preceding section. The intuition behind each technique that we propose has been detailed therein.

Simple CBR. A simple CBR [2] is a non-distance-weighting algorithm that uses a constant voting power function. Presenting it in another fashion, it takes the majority vote for classification.

Diff-CBR. This technique [3] which has been shown to be much better than Simple CBR is a distance-weighting algorithm that uses a constant voting power function.

C1 CBR. A case in the case base which is part of a spam cluster would have mostly spam cases among its k-NN (and vice versa). Such a case surrounded by spam cases being among the k-NN of a test case, intuitively gives a higher confidence that the test case is part of or in the near vicinity of the spam cluster (and vice versa). There is a host of clustering algorithms which rely on finding elements with a dense neighborhood and use them as seed points for identifying clusters ([5],[6] and [7]). C1 CBR is a variation of

Simple-CBR which incorporates this belief in a straightforward manner. It is a non-distance-weighting algorithm in which the voting power of a case is the number of cases with the same label as the case (in question) among its k-NN. In order that no case be assigned a voting power of zero, we include the case itself among the k-NN neighbors of the case to compute its voting power.

C2 CBR. This is a variation of Diff CBR along the same lines as C1 CBR. It is a distance weighting algorithm, where the voting power function is exactly the same as in C1 CBR.

C3 CBR. C1 and C2 CBR are suspected to suffer from a serious drawback. Consider a dense spam cluster and a singleton point in an isolated area of the case base. All points in the spam cluster would get a voting power of k when k nearest neighbors are considered (unsurprisingly), and the singleton point would also get a voting power of k (surprisingly!) if all its k-NNs are spam (possibly, they are part of the dense cluster) although they are a considerable distance away compared to the former case. C3 CBR tries to rectify this problem by introducing an additional parameter, which we hereafter refer to as the radius. The voting power of a case in the case base is computed as 1 + (number of cases with the same label as the case in question, and which fall within a distance of radius from the case). The addition of 1, once again is to ensure that no case gets a zero voting power. This is a non-distance-weighting algorithm. We would like to clarify at this point that determining an optimal value for radius is a non-trivial task.

Better Voting Power Function. A bit of thought is more than sufficient to come up with the insufficiencies of the C3 CBR voting power function. Consider a highly noisy space where a spam case has 50 spam cases and 50 legitimate cases within its radius. On the contrary, consider a pure space which has a sparse cluster where a spam case has just 5 spam neighbors within its radius. Intuitively, the second case deserves a better voting power whereas the C3CBR voting power function assigns a voting power of 51 to the former and 6 to the latter; a huge disparity indeed. Although the frequency of such hostile cases have to be studied, the disparity introduced by the C3CBR voting power function is so high that we can't let it go unattended. In this context, we choose to lay down some of the more intuitive desiderata for a voting power function.

Assume that the total number of cases in the radius of the case in question be t, the number of cases with a matching label among them be m1 and those with mismatching labels among the t cases be m2. Firstly, the voting power function should be directly related to m1. Secondly, the voting power function should be directly related to t. Thirdly, it should be inversely related to m2. Given that t is (m1 + m2), one might reasonably argue that any two of the above relations should be sufficient. Such a function is highly non trivial. We propose a voting power function, hereafter referred to as BVPF which we define as follows.

$$BVPF(t,m1,m2) = (m1 - m2) * \log(t) / t$$

A closer look at the function would reveal that it favors dense areas compared to sparse ones. To illustrate the aspect, BVPF(70,50,20) > BVPF(7,5,2). Although it is easy to create a hostile situation to BVPF, we argue that a hostile situation for C3CBR function is much more probable than one for BVPF. In the remaining algorithms that we propose, we consistently use BVPF as the voting power function.

C4 CBR. This is a non-distance-weighting algorithm which uses the BVPF as the voting power function.

C5 CBR. This is the distance-weighting algorithm which uses BVPF as the voting power function.

C6 CBR. As mentioned earlier, determining an optimal value for radius is a non-trivial task. C6 CBR tries to make the process as much insensitive to the value of the radius parameter. We distort the confidence function a bit and redefine it as following:

$$Confidence\_X\_C6CBR_{radius=r}(T, k\text{-}NN) = Confidence\_X_{radius=r}(T, k\text{-}NN) + Confidence\_X_{radius=2*r}(T, k\text{-}NN) + \ldots + Confidence\_X_{radius=n*r}(T, k\text{-}NN)$$

$Confidence\_X_{radius=r}(T, k\text{-}NN)$ denotes Confidence_X(T, k-NN) computed with BVPF as the voting power function and r taken as the radius for the BVPF computations. We consistently set n = 5 (the number of terms in the right-hand-side of the above equation) in the course of our experiments with C6 CBR.

6. Performance Measures Used

We use various performance measures for evaluating the different techniques for spam filtering described above. *Spam precision* is the percentage of messages classified as spam that truly are. *Spam recall* (interchangeably referred to as spam accuracy) is the proportion of actual spam messages that are classified as spam. Non-spam messages are usually called solicited messages or legitimate messages. *Legitimate precision*, analogously, is the percentage of messages classified as legitimate that truly are. *Legitimate recall* (interchangeably referred to as legitimate accuracy) is the proportion of actual legitimate messages that are classified as legitimate [4]. The *total accuracy* is the total number of messages classified correctly. Intuitively, the severity of the error of classifying a spam message as legitimate is much less than the severity of classifying a legitimate message as spam. Taking these into account, we define an *error cost function* as the sum of the errors with differential weighting for the two kinds of errors. The obvious parameter to this cost function would be the difference in severities. It has been shown [1] that the cost of the latter error is 999 times that of the former in a setting where spam messages are blocked from the user. In a scenario where messages are just flagged as spam by the filter, the disparity in severity comes down to 9. We analyze the techniques with both values for the severity disparity parameter.

7. Experiments, Results and Implications

In this section, we walk through (in chronological order) the results of the various experiments that have been conducted. As is typical in any experiment in this context, we divide the corpus into the training set and the test set. The training set forms the case base (and hence is hereafter referred to, as the case base) and each message in the test set is classified by the CBR making use of the case base. Given that the entire corpus (and hence the test set too) is labeled, we can get a feel of the performance of the algorithms in this regard.

7.1 Performance of non-radius based techniques

As explained in the preceding section, Diff-CBR, Simple CBR, C1 and C2 CBRs don't require a radius parameter. We experimented with them on varying case base sizes. We present the spam precision and total accuracy charts.

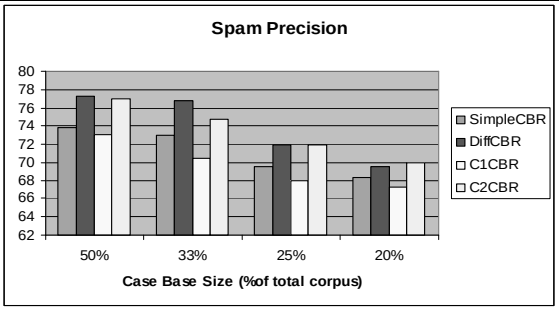


Figure 2. Spam Precision chart

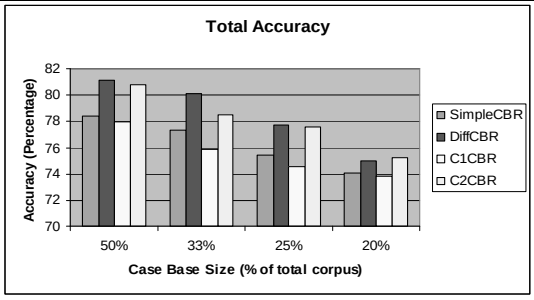


Figure 3. Total Accuracy Chart

As can be seen, Diff-CBR works the best in each of the cases, whereas C2 CBR does approach it closely in performance compared to other methods. Given that C2 CBR seems to be brushing shoulders with Diff-CBR and applies the common technique of distance weighting, we decided to look at the number of common errors that they make to gather an insight as to how much influence the voting power function actually had. The number of common errors was a surprisingly high 88% on the average which indicates that the performance of C2 CBR was more due to the distance-weighting component than the voting power function. As the comparison between C1 and Simple-CBR shows, the voting power function is actually worse than the constant power function used by the latter. Although these results are clearly disheartening, we choose to examine the extent of the effect caused by the drawback of the C1 and C2 voting power functions as mentioned in an earlier section.

7.2 C3CBR

We choose to analyze C3 CBR separately as all others to follow use the same voting power function. We chose to use a 50% case base, and increasing values of radius. We present the accuracy chart as below. As is evident from the results below, the accuracies don't approach that of the conventional techniques such as Diff and Simple CBR. But one interesting thing worth mentioning in this context is that, although C3 CBR accuracies are (slightly) lesser than Diff-CBR, the fraction of common errors

between C3 and Diff were 75% (compared to the figure of 88% for the C2-Diff pair) on the average. This gives us enough confidence that we are not searching in the dark and hence, we proceed to quantify the extent of the drawback discussed in the previous section.

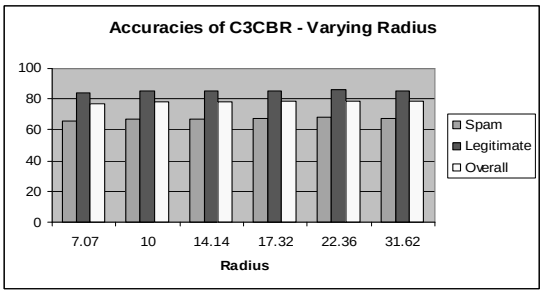


Figure 4. Accuracies for C3CBR

7.3 Performance of Techniques that use BVPF

In our experiments with C4, C5 and C6 CBR, we were able to arrive at significantly better results (with varying values of radius). Even C4 CBR, the non-distance weighting BVPF CBR, gave much better accuracies compared to earlier techniques. C5 and C6 CBRs performed exceedingly well in comparison with others on spam precision. Although we do not include the charts of all the experiments that were conducted, we present a list of representative charts to show the performances of each of the techniques discussed so far so as to assert that BVPF works exceedingly well as a voting power function. All these were done with 50% of the corpus as the case base.

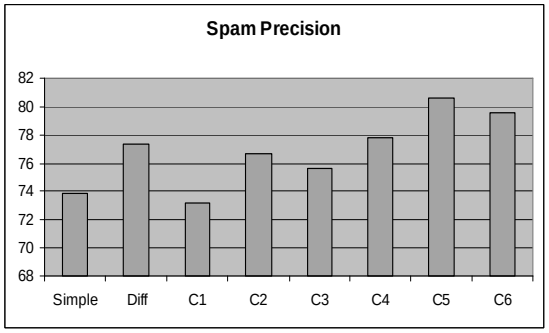


Figure 5. Spam Precision Chart

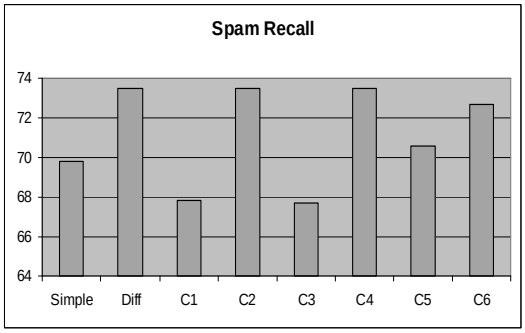


Figure 6. Spam Recall Chart

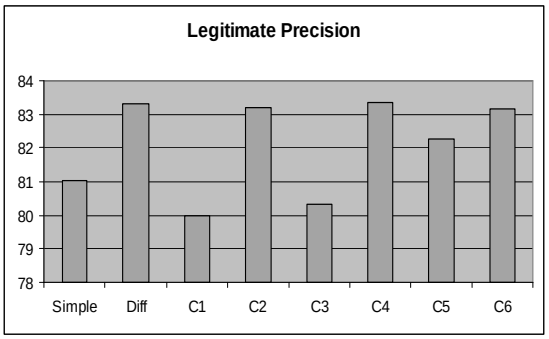


Figure 7. Legitimate Precision Chart

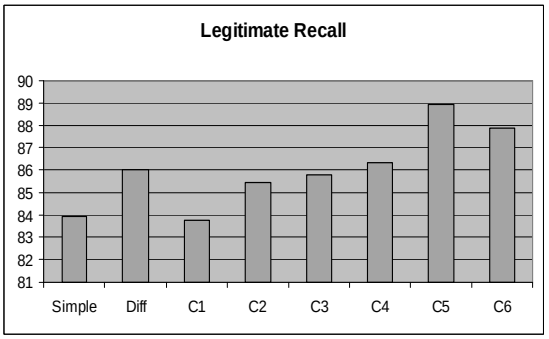


Figure 8. Legitimate Recall Chart

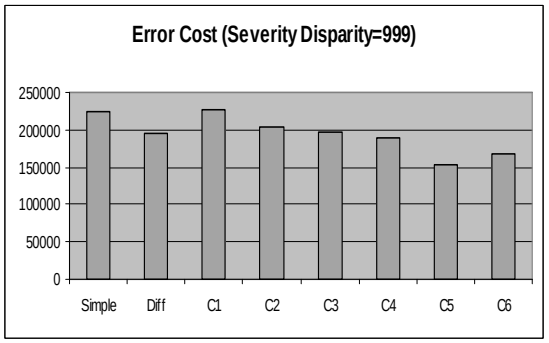


Figure 9. Error Costs (Svrty Disp = 999)

Figure 10. Error Costs (Svrty Disp = 9)

As can be seen, all techniques that use BVPF, viz., C4, C5 and C6 CBRs work much better than the others on the parameters that count the most, i.e., spam precision, legitimate recall, and hence error costs. C6 CBR performs better than Diff on all measures except for Spam Recall where Diff-CBR works slightly better. C5 CBR gives the lowest error costs, and gives the highest Spam Precision and Legitimate Recall. On the whole, C5 and C6 techniques are way ahead of the conventional techniques. This proves our point that making use of the patterns in the case base is does improve performance very much.

8. Voting Powers and a Dynamic Case Base

Given that all our experiments have been on static case bases, it is reasonable enough to devote a section on how the computation of voting powers would be in a real-world scenario where cases get added and deleted from the case base. We provide a straightforward method to update the voting powers when a case gets added or deleted

from the case base. We describe an algorithm to show how the BVPF powers can be updated on addition of a case base and omit other details as they would be a straightforward modification of the algorithm. We propose storing the <m1, m2, BVPF power> triplets for each case in the case base. The linear algorithm described below updates these triplets for relevant cases in the case base when a case gets added.

Table 3. Updating Voting Powers on Case Addition

Algorithm Update_On_Addition

Update_On_Addition(NewCase n, Case Base C){
m1 = m2 = 0;
for(each case c in C){
if(distance(c,n) < radius)
if(label(c) == label(n))
increment m1; increment m1 of n and re-compute BVPF of n;
else
increment m2; increment m2 of n and re-compute BVPF of n;
}
Store <m1,m2,BVPF(m1+m2,m1,m2)> for the new case n;
}

9. Contributions and Future Work

We have, by means of this paper, provided approaches to make use of the patterns in the case base by means of associating each case with a voting power to improve spam filtering using CBR. This is, to the best of our knowledge, the first work on making use of the skewed distribution in the case base for a classification task. We have laid down the concerns on the design of a voting power function. Further, we have experimented exhaustively and made the implications of the voting power functions explicit. As a part of future work in this regard, we propose to look deeper into the BVPF function and hostile cases to it. Further, as mentioned earlier, BVPF favors dense clusters over sparser ones. The implications of such a bias have to be examined in detail. BVPF is just our first approach in satisfying the desiderata for a voting power function and we have to look to find better variants of BVPF. Secondly, clustering is a data mining task which has been

receiving a lot of attention of late. We would like to explore the feasibility of actually clustering the case base and making use of the clusters for the CBR classification task at hand. Further, we would like to look into other domains and test the applicability of BVPF and variants for classification tasks therein.

References

1. Androutsopoulos, Paliouras, Karkaletsis, Sakkis, Spyropoulos and Stamatopoulos, 2000, *Learning to filter spam e-mail: a comparison of a naive Bayesian and a memory-based approach*, PKDD Workshop on Machine Learning and Textual Information Access, 2000

2. Cunningham, Nowlan, Delany and Haahr, 2003, *A case-based approach to spam filtering that can track concept drift*, Proceedings of the ICCBR Workshop on Long-Lived CBR Systems, Norway, 2003

3. Sakkis, Androutsopoulos, Paliouras, Karkaletsis, Spyropoulos and Stamatopoulos, 2003, *A memory-based approach to anti-spam filtering for mailing lists*, Journal of Information Retrieval, Kluwer, 2003

4. Sahami, Dumais, Heckerman & Horvitz, 1998, *A bayesian approach to filtering junk e-mail*, AAAI-98 Workshop on Learning for Text Categorization, 1998

5. Ester, Kriegel, Sander, Xu, 1996, *A Density based algorithm for discovering clusters in large spatial databases with noise*, International Conference on Knowledge Discovery in Databases, KDD-1996

6. Hinneburg and Keim, 1998, *An efficient approach to clustering in large multimedia databases with noise*, International Conference on Knowledge Discovery in Databases, KDD-1998

7. Ankerst, Breunig, Kriegel and Sander, 1999, *OPTICS: Ordering Points to Identify the Clustering Structure*, Proceedings of the ACM SIGMOD Conference

Chapter 5

Outputs from the Project So Far

Accepted Publications

"Corpus Based Unsupervised Labeling of Documents", Accepted at the 19th International FLAIRS Conference, May 2006, Florida (with Delip Rao and Dr. Deepak Khemani)

"Differential Voting in Case Based Spam Filtering", Accepted at the Industrial Conference on Data Mining (ICDM Leipzig 2006), Leipzig, July 2006 (with Delip Rao and Dr. Deepak Khemani)

Communicated Publications

"Unsupervised Learning From URL Corpora"

"Building Clusters of Related Words: An Unsupervised Approach"